Additively homomorphic ring-LWE masking

Reparaz, Oscar; de Clercq, Ruan; Roy, Sujoy Sinha; Vercauteren, Frederik; Verbauwhede, Ingrid

DOI:
10.1007/978-3-319-29360-8_15

Additively Homomorphic ring-LWE Masking

Oscar Reparaz, Ruan de Clercq, Sujoy Sinha Roy,
Frederik Vercauteren and Ingrid Verbauwhede

COSIC/KU Leuven and iMinds
Kasteelpark Arenberg 10, B-3001 Leuven, Belgium
firstname.lastname@esat.kuleuven.be

Abstract. In this paper, we present a new masking scheme for ring-LWE decryption. Our scheme exploits the additively-homomorphic property of the existing ring-LWE encryption schemes and computes an additive-mask as an encryption of a random message. Our solution differs in several aspects from the recent masked ring-LWE implementation by Reparaz et al. presented at CHES 2015; most notably we do not require a masked decoder but work with a conventional, unmasked decoder. As such, we can secure a ring-LWE implementation using additive masking with minimal changes. Our masking scheme is also very generic in the sense that it can be applied to other additively-homomorphic encryption schemes.

1 Introduction

Most public-key cryptography deployed today will not withstand attacks by a quantum computer. Shor's algorithm [Sho99] can break RSA, discrete logarithms and elliptic-curve cryptography in polynomial time using a quantum computer. The National Security Agency (NSA) has recently announced that quantum computing is a threat to the existing public key infrastructure, and has recommended a transition to quantum resistant public key algorithms [nsa15]. In recent years significant progress was made to improve public-key cryptosystems based on computational problems that will remain secure even in the presence of powerful quantum computers. Regev's *learning with errors* (LWE) problem [Reg05] and its ring variant, known as the *ring*-LWE problem have become very popular in designing public key encryption, key exchange, digital signature and homomorphic encryption schemes. Several recent publications such as [PG14, PDG14, RVM+14, GOPS13, RVV14, dCRVV15, APS13, LSR+15, BSJ15, POG15] show that ring-LWE based encryption and digital signature schemes are faster and relatively easier to implement compared to elliptic curve cryptography (ECC) algorithms.

Though secure against quantum computing, ring-LWE based cryptography offers no inherent protection against side-channel

attacks [Koc96]. It is well-known that a vanilla, unprotected implementation of a cryptographic algorithm running on an embedded device can be broken if the adversary can observe a side-channel, such as the instantaneous power consumption, the EM radiation or some timing information. A particularly effective method to extract secrets, such as cryptographic keys or passwords, from embedded devices is Differential Power Analysis (DPA) [KJJ99].

Masking [CJRR99, GP99] is a provable sound countermeasure against DPA. First-order masking works by probabilistically splitting every intermediate into two shares such that each share is statistically independent from the intermediate. This property ought to preserve through the entire computation. A masking scheme defines how the computation on masked data should be performed. Masking, of course, comes at a cost. Masked implementations incur area, time and energy overheads. In public-key cryptosystems, the decryption operation is normally the prime target for DPA protections, as it is the component that manipulates long-term secrets.

Post-quantum cryptosystems are not yet as mature as RSA, Diffie-Hellman or ECC. There is ongoing research to determine the exact security offered by a concrete parameter choice, to determine which padding schemes should be used, to design fast and memory-efficient implementations that can compete with classical public-key cryptography and to write protected implementations against side-channel analysis.

A first step in a masked ring-LWE implementation is the work [RRVV15], hereafter refered to as the CHES 2015 approach. This approach takes an unmasked ring-LWE processor and adds masking with a bespoke, customized masked decoder. The overhead is roughly 2.6 times more cycles and the impact in area is very small.

Our contribution. In this paper we propose a new masking scheme to protect the secret key during decryption operations in ring-LWE cryptosystems. Our masking scheme is based on the additively homomorphic nature of the existing ring-LWE encryption. A mask is computed by encrypting a random message and then the mask is added to the ciphertext. This operation randomizes the ciphertext and mitigates the side-channel leakage problem.

Our solution has the advantage compared to the CHES 2015 approach that we do not require additional hardware (nor software) to compute the final decoding operation. The masking scheme is applicable to both hardware and software implementations. A caveat of our approach is that

we need to place additional assumptions on the underlying arithmetic hardware compared to the CHES 2015 approach.

2 Background

For a complete view of the system, we describe the entire ring-LWE cryptosystem. In this paper we focus on the DPA security of the ring-LWE decryption operation.

Notation. We denote by $R = \mathbb{F}_q[x]/(f(x)), +, *$ a modular polynomial ring over base field $\mathbb{F}_q$. When we want to access a specific coefficient of a polynomial s we write $s[i]$. The operation $\oplus$ is the xor operation on bits or strings of bits.

Review of ring-LWE based encryption scheme. In the literature there are several encryption schemes based on the ring-LWE problem, for example [LPR10], [FV12], [BLLN13] etc. The major algorithms in these encryption schemes are: key-generation, encryption and decryption. These algorithms perform message-encoding, discrete Gaussian sampling, polynomial addition/subtraction/multiplication, and decoding as the primitive operations.

In this paper, we use the scheme proposed by Lyubashevsky, Peikert, and Regev (LPR) [LPR10]. Though our masking scheme is generic and works with the other ring-LWE encryption schemes, we choose the LPR scheme for the analysis mainly due to the availability of several efficient implementations [PG14, RVM+14, GOPS13, dCRVV15, LSR+15, POG15] and due to the existence of a DPA resistant masked implementation [RRVV15].

The three main operations in the LPR encryption scheme are described below. The parameters are (n, q, σ) where n is the dimension of polynomial ring, q is the modulus and σ is the standard deviation of the discrete Gaussian distribution.

- *Key generation.* Two polynomials r and s are generated by sampling the coefficients from the discrete Gaussian distribution. Next a new polynomial $p = r - g * s$ is computed where g is a globally known base polynomial. The key generation outputs s as the secret key and p as the public key.
- *Encryption.* The n-bit input plaintext is encoded as a ring element $\bar{m} \in R$ by multiplying the bits by $q/2$. The encryption operation generates three error polynomials e_1, e_2 and e_3 using the discrete

Gaussian sampler. These error polynomials are used as noise. The ciphertext is a pair of polynomials (c_1, c_2) where $c_1 = g * e_1 + e_2$ and $c_2 = p * e_1 + e_3 + \bar{m}$.

– *Decryption.* In the decryption phase s is used to compute the intermediate message $\tilde{m} = c_1 * s + c_2$. This intermediate plaintext contains noise. Next, a decoding is performed to recover the original plaintext bits: $m_{\text{recovered}} = \mathit{decode}(\tilde{m})$. The simplest decoder just compares each coefficient of $\tilde{m}$ with $q/2$: if the distance is small (i.e. $< q/4$) it returns 1 otherwise it returns 0.

Among all the computations, polynomial multiplication is the costliest. Most of the reported implementations use the Number Theoretic Transform (NTT) to accelerate the polynomial multiplications. In the implementation in [RVM+14] the ciphertext is kept in the NTT domain to reduce the number of NTTs and inverse NTTs (INTTs). When c_1, c_2 and s are in the NTT domain, the plaintext bits are computed as $m_{\text{recovered}} = \mathit{decode}\big(\text{INTT}(c_1 \cdot s + c_2)\big)$. Here $\cdot$ is the coefficient-wise multiplication operator.

Review of CHES2015 approach. The paper [RRVV15] proposes to mask the ring-LWE decryption by additively splitting the secret s into two shares s', s'' such that $s = s' + s''$. The masked decryption proceeds as follows: it first computes one branch

$$a' = \text{INTT}(c_1 \cdot s' + c_2), \tag{1}$$

then proceeds with the computation of the second branch:

$$a'' = \text{INTT}(c_1 \cdot s'') \tag{2}$$

and finally outputs the pair of the mask bit and the masked message bit $(m', m'') = \text{masked-decoder}(a', a'')$.

The random splitting of s into two shares s' and s'' works as a countermeasure against DPA during the coefficient-wise multiplications. The main difficulty is the masked-decoder block. This block performs the threshold th computation in the masked domain, yielding Boolean masked results m' and m''. Inside the decoder block, the two input shares a' and a'' are compared with a lookup table to check if a set of rules is satisfied or not. When the rules are not satisfied, the shares are refreshed by adding and subtracting a small refreshment-value Δ with the two shares, and then checking the rules again. The masked decoder implementation in [RRVV15] performs the refreshing operation 16 times in order to achieve constant time decoding with high success probability.

3 Additively Homomorphic ring-LWE Masking

Core idea. The central idea is that the LPR encryption scheme presented in Section 2 is additively homomorphic. This means that for any two ciphertexts (c_1, c_2) and (c'_1, c'_2) corresponding to the respective encryptions of m and m' under the same public key, $(c_1 + c'_1, c_2 + c'_2)$ will be an encryption of $(m \oplus m')$. Hence we can write the following equation:

$$\text{decryption}(c_1, c_2) \oplus \text{decryption}(c'_1, c'_2) = \text{decryption}(c_1 + c'_1, c_2 + c'_2) \quad (3)$$

This additive homomorphism can be exploited to randomize the computation of the decryption operation. The randomization technique is explained below.

The proposed randomized decryption. To perform the decryption of (c_1, c_2) in a randomized way, the implementation follows the following steps:

1. Internally generate a random message m' unknown to the adversary
2. Encrypt m' to (c'_1, c'_2)
3. Perform decryption$(c_1 + c'_1, c_2 + c'_2)$ to recover $m \oplus m'$.

The masked recovered message is the tuple $(m', m \oplus m')$.

This approach has the nice property of not requiring a masked decoder. One can use an unprotected decoder function. The obvious disadvantage is that extra circuitry or code is required to perform the encryption. Another disadvantage is the increased decryption failure rate. When two ciphertexts are added, the amount of noise increases. The added noise increases the decryption failure rate as we will see in Section 4.3.

4 Discussion

4.1 Analysis

First-order DPA. Our countermeasure can be thought of as ciphertext blinding. Note that there is no attacker-known, nor attacker-controlled inputs that are mixed with the secret key s. Thus, straightforward first-order DPA attack does not immediately apply. Nevertheless, more refined first-order DPA attacks do apply.

First-order attacks. Note that the key is not masked. Thus, we do not claim theoretic first-order security. Our randomization makes it harder for the attacker to model the power consumption (and thus harder to DPA). In Appendix A we describe a strategy to detect whether $s[i] = 0$ or $s[i] \neq 0$, which leads to an entropy loss. This seems not to significantly affect security for the following reason. First, remember that s is handled in the NTT domain, so that the probability of the event $s[i] = 0$ is $1/q$. If there are w coefficients for which $s[i] = 0$, the dimension is effectively reduced by w. Since $q > n$, we expect w to be very small and thus not to lose much in the dimension of the system. The same effect can occur at a smaller scale, exploiting intermediates from within the multiplication. In this situation, the consequences are more serious. Therefore, the underlying hardware must ensure that intermediates from inside the multiplication are noisy enough to be hard to exploit in this way.

4.2 Comparison with previous work

In this section we compare our solution with the CHES 2015 approach.

Offline precomputations. Our solution allows to precompute the encryption of m' into (c'_1, c'_2). This follows since m' is independent from the message m to be decrypted. In contrast, the CHES 2015 approach does not allow to precompute any of the values from Eq. 1 nor Eq. 2. This potential precomputation minimizes the impact of the countermeasure on the running time, as detailed in the next section.

Simplicity. The implementation complexity of our solution is remarkably low, both in software or hardware. In comparison, the CHES 2015 approach would need a careful implementation of the masked decoder block. This block is delicate to implement. In particular, the practitioner should pay careful attention to leaking distances if implemented in software, since during the masked decoding both shares are handled in contiguous temporal locations. In hardware comparable observations apply during the implementation of the masked tables.

In contrast, our approach is very easy to implement. The implementation handles both shares of all intermediates far from each other, minimizing the possibility of unintended interferences between shares (and thus first-order leaks).

Is the masked decoder needed? In this paragraph we would like to point out an important difference between the CHES 2015 approach and the

one presented in Section 3. Namely, in this paper we do not require a masked decoder, while the CHES 2015 solution does. One can wonder if the masked decoder of the CHES 2015 approach is really needed: after all, Eq. 3 may seem to imply that the decoding function is linear. However, this is clearly not the case.

The difference is that, in our additively-homomorphic masking scheme the inputs to the decoder are coefficients resulting from the *proper* decryption with respect to the secret key s, and hence the input coefficients are distributed around 0 or $q/2$. Whereas in the CHES 2015 approach, the shared coefficients a' and a'' in (Eq. 1 and Eq. 2) are not *individually* proper decryptions of a valid message; and hence are uniformly distributed in $(\mathbb{F}_q, \mathbb{F}_q)$. This is why the CHES 2015 requires a custom decoder, whereas our masking scheme does not.

4.3 Error rates

The LPR encryption scheme is probabilistic in nature, i.e. the decryption of a valid ciphertext may produce an incorrect plaintext with a small probability. A decryption failure occurs when the noise in the coefficient-to-be-decoded exceeds the threshold value of the decoder. In our additive masking scheme the addition of two ciphertexts also adds the noises present in the two ciphertexts: the error coefficients in the new ciphertext could be at most one bit larger than the error coefficients in the two ciphertexts. This larger noise increases the decryption failure rate. To know the exact decryption failure rate we performed experiments for the parameter set $(n, q, \sigma) = (256, 7681, 4.51)$ [GFS+12] corresponding to a medium-level security. The parameter set was used in [PG14, RVM+14, dCRVV15, LSR+15, POG15] to implement encryption schemes. When the masking is turned off, the decryption failure rate is 3.6×10^{-5} per bit. The failure rate increases to 3.3×10^{-3} per bit when the masking turned on.

The increase in the decryption failure rate can be compensated at the cost of a minor deterioration in the security by using the techniques as follows.

- The modulus q can be increased by one bit. This increment in the size of q (from 13 bits to 14 bits) does not slow down our software implementation since the underlying processor architecture is 32 bit, and hence the processing times for both 13 and 14 bit coefficients are the same.
- As suggested by one of the anonymous reviewers, decreasing the standard deviation σ of the discrete Gaussian distribution may be

more effective than increasing the size of q as the final noise is in the order of σ^2.

5 Implementation results

The presented masking scheme is suitable for implementation both in hardware and software. We wrote a reference version of the proposed countermeasure in C99. The implementation follows the same lines as de Clercq et al. [dCRVV15].

Overheads. The overhead of our solution with respect to an unprotected decryption is one random message generation, one extra encryption and one coefficient addition. This incurs a negligible code size increase if the encryption operation is available. In terms of speed, the costliest process is the encryption. It is 2.8 times slower than the decryption. However, this computation can be performed in advance before even knowing the ciphertext to be decrypted.

6 Experimental results

In this section we describe the key-recovery DPA attacks on an ARM Cortex-M4 processor that we performed to assess the security of our solution.

Experimental setup. We compiled the reference implementation with `arm-none-eabi-gcc` version 4.8.4 20140526 without any special optimization flags (note that we do not aim at maximum speed or code efficiency). We flashed an STM32F407VGT6 microcontroller featuring an ARM Cortex-M4 core running at 168 MHz (full speed) and an RNG that "delivers 32-bit random numbers generated by an integrated analog circuit"[1]. We collected contactless power measurements by placing a Langer LF-R 400 magnetic field probe in the vicinity of the chip power supply circuitry as indicated in Figure 6. Traces are synchronized by a GPIO pin.

Methodology. We follow a standard methodology to assess the security of our countermeasure. We first attack our implementation when the source of randomness is switched off—that is, the whole computation is deterministic. This is equivalent to switching off the countermeasure.

[1] http://www.st.com/web/en/resource/technical/document/datasheet/DM00037051.pdf

Therefore, attacks are expected to work against this mode of operation. Nevertheless, successful attacks in this scenario serve to confirm that the experimental setup is indeed sound. In the second part of our analysis we switch on the randomness to observe security gained exclusively by the countermeasure.

We assume that when the adversary places hypotheses on certain key coefficients, he knows all other key coefficients. This allows the adversary to easily predict intermediates deep into the computation. This adversarial model may seem quite strong; however, due to the mathematical structure of the scheme it is possible to predict deep intermediates with low effort.

An overview EM trace is depicted in Figure 1. The trace spans the entire protected computation as described in Section 3. Features of this EM trace are more visually recognisable in the cross-correlation picture of Figure 2. We can recognize the two most time-consuming blocks: the encryption of m' and the subsequent computation of decryption$(c_1 + c'_1, c_2 + c'_2)$.

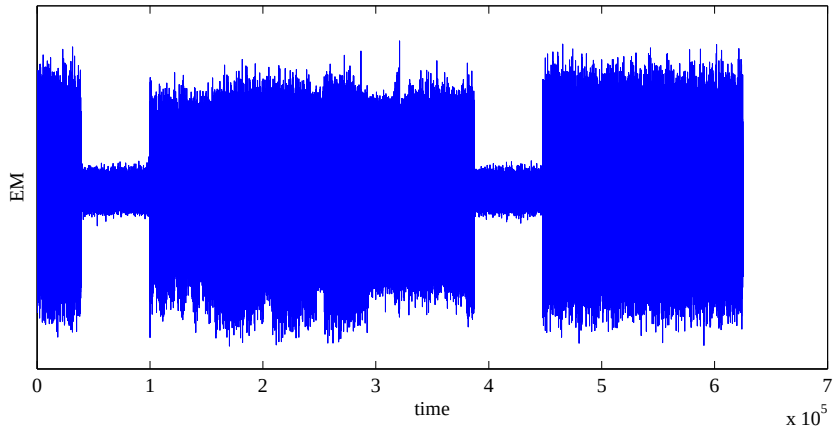

Fig. 1: An exemplary EM trace covering the whole protected decryption. Data series with large number of samples are difficult to plot; patterns are more visible with other plotting techniques cf. Figure 2.

Masks off. We modeled the power consumption of a 32-bit register holding the result of a `MUL` instruction as the Hamming distance between two consecutive values, and applied standard CPA [BCO04]. When the randomization is switched off the CPA attack is successful. In this scenario

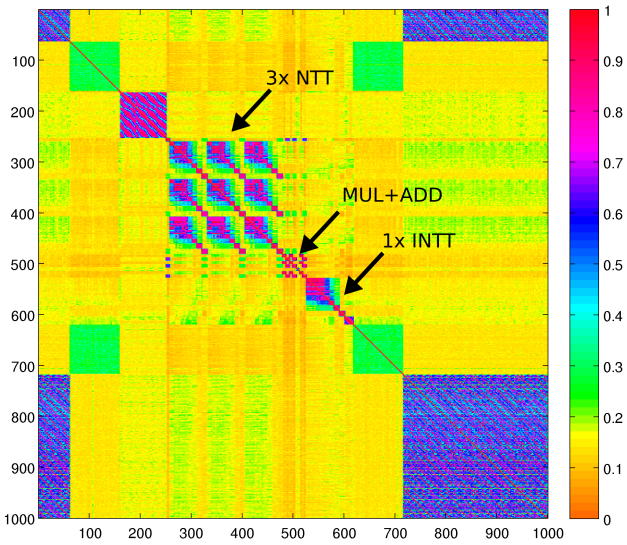

Fig. 2: Cross-correlation of a single trace.

the adversary learns the "random" values, he can predict any intermediate, and thus the attack is expected to work. Nevertheless, this confirms that the setup is sound.

Figure 3 shows the result of correlating 5 000 traces against predictions of an intermediate that appears towards the end of the INTT computation. Note that there are plenty of time samples that allow key-recovery; this is because this intermediate is handled at many other times during the execution of the decryption block.

The evolution of the Pearson's correlation coefficient as the number of traces increases is plotted in Figure 4. We can see that starting from 1 000 measurements the attack is successful.

Masks on. We repeated the same procedure when the randomness is switched on. This is equivalent to activating the countermeasure. At the time of this writing, we had available 5 000 traces. as Figure 5 shows. The countermeasure makes harder the DPA attack: the correlation for the correct key hypothesis does not stand out among other key hypothesis. We acknowledge that it is suspiciously high. A more detailed study is planned for the extended version of this paper.

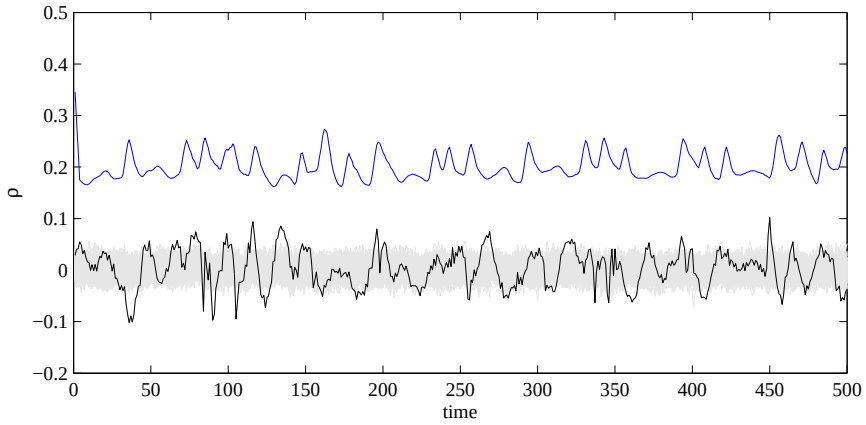

Fig. 3: Top: EM trace in the region where the modular multiplication is performed. The time axis spans around 10 instructions, including `MUL.W`. Bottom: CPA results. Correct key coefficient hypothesis in black; incorrect hypotheses in grey. Masks off.

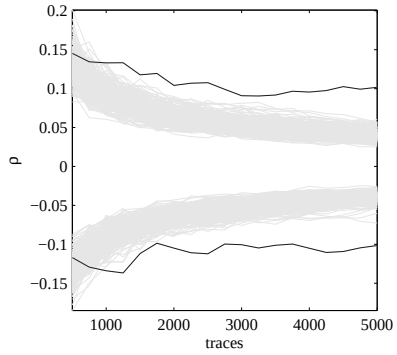

Fig. 4: Evolution of CPA results masks off.

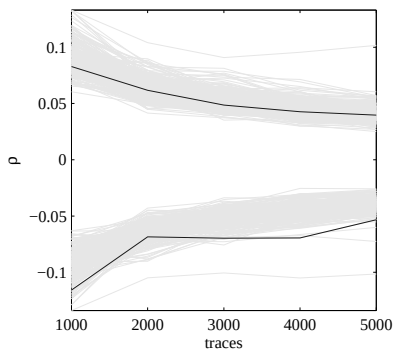

Fig. 5: Evolution of CPA results masks on.

7 Conclusion

In this paper we proposed a new masking scheme for protecting ring-LWE decryption against differential power analysis based attacks. The proposed masking technique is more generic than the state of the art and

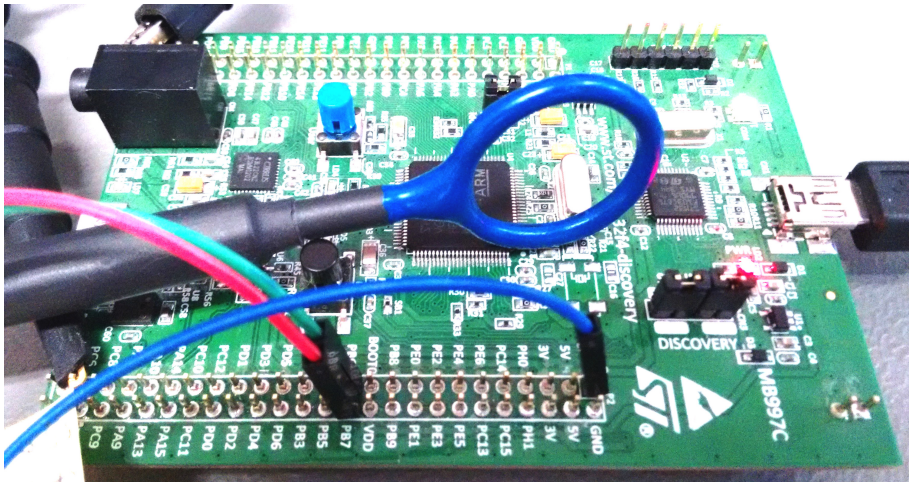

Fig. 6: Setup photography showing the orientation of the H-field pick-up probe.

can be applied to all ring-LWE encryption schemes that are additively homomorphic. Moreover we showed that the masking scheme is easy to implement and does not require any masked decoder circuit or software.

Acknowledgements. The authors would like to thank the PQCrypto 2016 reviewers for their valuable comments. This work has been supported in part by the European Commission through the ICT programme under contracts H2020-ICT-645622 PQCRYPTO, H2020-ICT-644209 HEAT and FP7-ICT-2013-10-SEP-210076296 PRACTICE; by the Research Council KU Leuven TENSE (GOA/11/007); by the Flemish Government FWO G.0550.12N, G.00130.13N and G.0876.14N; and by the Hercules Foundation AKUL/11/19. Oscar Reparaz is funded by a PhD fellowship of the Fund for Scientific Research - Flanders (FWO). Sujoy Sinha Roy was supported by Erasmus Mundus PhD Scholarship.

References

APS13. Aydin Aysu, Cameron Patterson, and Patrick Schaumont, *Low-cost and Area-efficient FPGA Implementations of Lattice-based Cryptography*, HOST, 2013, pp. 81–86.

BCO04. Eric Brier, Christophe Clavier, and Francis Olivier, *Correlation power analysis with a leakage model*, CHES, LNCS, vol. 3156, Springer, 2004, pp. 16–29 (English).

BLLN13. Joppe W. Bos, Kristin Lauter, Jake Loftus, and Michael Naehrig, *Improved security for a ring-based fully homomorphic encryption scheme*, Cryptology ePrint Archive, Report 2013/075, 2013, `http://eprint.iacr.org/`.

BSJ15. Ahmad Boorghany, Siavash Bayat Sarmadi, and Rasool Jalili, *On constrained implementation of lattice-based cryptographic primitives and schemes on smart cards*, ACM Trans. Embed. Comput. Syst. **14** (2015), no. 3, 42:1–42:25.

CJRR99. Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi, *Towards sound approaches to counteract power-analysis attacks*, CRYPTO, LNCS, vol. 1666, Springer, 1999, pp. 398–412 (English).

dCRVV15. Ruan de Clercq, Sujoy Sinha Roy, Frederik Vercauteren, and Ingrid Verbauwhede, *Efficient software implementation of ring-lwe encryption*, Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition, DATE 2015, Grenoble, France, March 9-13, 2015 (Wolfgang Nebel and David Atienza, eds.), ACM, 2015, pp. 339–344.

FV12. Junfeng Fan and Frederik Vercauteren, *Somewhat practical fully homomorphic encryption*, Cryptology ePrint Archive, Report 2012/144, 2012, `http://eprint.iacr.org/`.

GFS+12. Norman Göttert, Thomas Feller, Michael Schneider, Johannes Buchmann, and Sorin Huss, *On the design of hardware building blocks for modern lattice-based encryption schemes*, CHES, LNCS, vol. 7428, Springer, 2012, pp. 512–529 (English).

GOPS13. Tim Güneysu, Tobias Oder, Thomas Pöppelmann, and Peter Schwabe, *Software Speed Records for Lattice-based Signatures*, Post-Quantum Cryptography, Springer, 2013, pp. 67–82.

GP99. Louis Goubin and Jacques Patarin, *DES and differential power analysis the duplication method*, CHES, LNCS, vol. 1717, Springer, 1999, pp. 158–172 (English).

GT02. Jovan Dj. Golic and Christophe Tymen, *Multiplicative masking and power analysis of AES*, Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers (Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, eds.), Lecture Notes in Computer Science, vol. 2523, Springer, 2002, pp. 198–212.

KJJ99. Paul Kocher, Joshua Jaffe, and Benjamin Jun, *Differential power analysis*, CRYPTO, LNCS, vol. 1666, Springer, 1999, pp. 388–397.

Koc96. Paul Kocher, *Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems*, CRYPTO, LNCS, vol. 1109, Springer, 1996, pp. 104–113 (English).

LPR10. Vadim Lyubashevsky, Chris Peikert, and Oded Regev, *On ideal lattices and learning with errors over rings*, EUROCRYPT, LNCS, vol. 6110, Springer, 2010, Full Version available at Cryptology ePrint Archive, Report 2012/230, pp. 1–23 (English).

LSR+15. Zhe Liu, Hwajeong Seo, Sujoy Sinha Roy, Johann Großschädl, Howon Kim, and Ingrid Verbauwhede, *Efficient ring-lwe encryption on 8-bit avr processors*, Cryptology ePrint Archive, Report 2015/410, 2015, `http://eprint.iacr.org/`.

nsa15. *Cryptography today*, Last Modified on Aug 19, 2015, `https://www.nsa.gov/ia/programs/suiteb_cryptography/index.shtml`.

PDG14. Thomas Pöppelmann, Léo Ducas, and Tim Güneysu, *Enhanced Lattice-Based Signatures on Reconfigurable Hardware*, Cryptographic Hardware and Embedded Systems CHES 2014, vol. 8731, 2014, pp. 353–370.

PG14. Thomas Pöppelmann and Tim Güneysu, *Towards practical lattice-based public-key encryption on reconfigurable hardware*, Selected Areas in Cryptography – SAC 2013, LNCS, vol. 8282, Springer, 2014, pp. 68–85.

POG15. Thomas Pöppelmann, Tobias Oder, and Tim Güneys, *High-performance ideal lattice-based cryptography on 8-bit atxmega microcontrollers*, Cryptology ePrint Archive, Report 2015/382, 2015, `http://eprint.iacr.org/`.

Reg05. Oded Regev, *On lattices, learning with errors, random linear codes, and cryptography*, Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing (New York, NY, USA), STOC '05, ACM, 2005, pp. 84–93.

RRVV15. Oscar Reparaz, Sujoy Sinha Roy, Frederik Vercauteren, and Ingrid Verbauwhede, *A masked ring-lwe implementation*, Cryptographic Hardware and Embedded Systems - CHES 2015 - 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings (Tim Güneysu and Helena Handschuh, eds.), Lecture Notes in Computer Science, vol. 9293, Springer, 2015, pp. 683–702.

RVM+14. Sujoy Sinha Roy, Frederik Vercauteren, Nele Mentens, Donald Donglong Chen, and Ingrid Verbauwhede, *Compact Ring-LWE Cryptoprocessor*, Cryptographic Hardware and Embedded Systems CHES 2014, vol. 8731, 2014, pp. 371–391.

RVV14. Sujoy Sinha Roy, Frederik Vercauteren, and Ingrid Verbauwhede, *High Precision Discrete Gaussian Sampling on FPGAs*, Selected Areas in Cryptography–SAC 2013 (2014), 383–401.

Sho99. Peter Williston Shor, *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*, SIAM Review **41** (1999), 303–332.

A An attack on the multiplication

An adversary could mount the following attack with a zero-value power model to recover only whether $s[i] = 0$ or not. Note that the distribution of $(c_1 + c_1') \cdot s$ when $s = 0$ and $c_1 + c_1'$ is uniform random is different from the distribution of $(c_1 + c_1') \cdot s$ when $s \neq 0$. This effect resembles [GT02], with the important difference that here the attacker has no control over $(c_1 + c_1')$ and that the outcome of the attack is recovering only whether $s[i] = 0$ or not.

1. locate time samples where $(c_1 + c_1')[i] \cdot s[i]$ is handled $i \in \{0, \ldots, 255\}$.
2. cluster $(c_1 + c_1')[i] \cdot s[i]$ into two groups according to mean power consumption (or variance).
3. tag the two groups as $s[i] = 0$ or $s[i] \neq 0$.